AMENDMENTS TO THE CLAIMS

Claims 1-6, 8-21, and 23 were originally pending. Please amend claims 16-17, 19-21, and 23. Kindly cancel claim 12. No claims have been added. Accordingly, claims 1-6, 8-11, 13-21, and 23 remain pending. The following listing of claims replaces all prior versions, and listings of claims in the application.

Listing of Claims:

1. (Previously presented) A method to be implemented in a computer system comprising a processor and a memory, the method for managing a run queue comprising a first plurality of threads sorted with respect to one another based on thread priority, the method comprising:

in a deterministic amount of time equivalent to an amount of time to insert a single thread into the run queue, associating a second plurality of threads that is priority sorted with the run queue in a manner that maintains a priority based scheduling semantic of the run queue.

2. (Previously presented) A method as recited in claim 1, wherein the second plurality of threads comprises a root thread, and wherein associating the second plurality of threads with the run queue further comprises inserting only the root thread into the run queue.

3. (Previously presented) A method as recited in claim 1, wherein the associating the second plurality of threads with the run queue further comprises inserting each thread in the second plurality of threads into the run queue independent of any additional other queue access.

4. (Previously presented) A method as recited in claim 1, wherein associating the second plurality of threads with the run queue further comprisesinserting only a root thread of the second plurality of threads into the run queue .

5. (Previously presented) A method as recited in claim 1, wherein associating the second plurality of threads with the run queue further comprises:

inserting only a root thread of the second plurality of threads into the run queue; and

wherein the method further comprises:

removing the root thread from the run queue; and

responsive to removing the root thread, inserting a next thread of the second plurality of threads into the run queue such that the priority based scheduling semantic of the run queue is preserved.

6. (Previously presented) A method as recited in claim 1, wherein the method further comprises:

inserting a root thread of the second plurality of threads into the run queue;

removing the root thread from the run queue for execution; and

responsive to removing the root thread and independent of any additional other queue access, inserting a next thread of the second plurality of threads into the run queue.

7. (Canceled).

8. (Previously presented) A system for managing a run queue, the run queue comprising a first plurality of threads, each thread in the first plurality of threads having a respective priority, the first plurality of threads being sorted such that a thread having a high priority is removed from the run queue before a thread having a lower priority, the system comprising:

a memory for storing the run queue and computer-executable instructions;

a processor operatively coupled to the memory, the processor being configured to execute the computer–executable instructions for:

in an amount of time to insert a single thread into the run queue, associating the second plurality of threads that is priority sorted with the run queue, the associating maintaining a priority based scheduling semantic of the run queue.

9. (Previously presented) A system as recited in claim 8, wherein associating the second plurality of threads with the run queue is performed independent of more than a single other queue access.

10. (Previously presented) A system as recited in claim 8, wherein the second plurality of threads comprises a root thread operatively coupled to one or more other threads of the second plurality of threads, each of the one or more other threads having a respective priority that is a lower priority or an equal priority as compared to a priority of the root thread.

11. (Previously presented) A system as recited in claim 8, wherein associating the second plurality of threads with the run queue further comprises inserting only a root thread of the second plurality of threads into the run queue.

12. (Canceled).

13. (Previously presented) A system as recited in claim 8:

wherein the first plurality of threads is a first linked list data structure;

wherein the second plurality of threads is a second linked list data structure comprising a root node that is operatively coupled to one or more other threads in the second plurality of threads; and

wherein the single insert operation is an operation comprising inserting the root node into a position in the first linked list data structure.

14. (Previously presented) A system as recited in claim 8, wherein associating the second plurality of threads with the run queue further comprises:

inserting only a root thread of the second plurality of threads into the run queue; and

wherein the method further comprises:

removing the root thread from the run queue; and

responsive to removing the root thread, inserting a next thread of the second plurality of threads into the run queue such that a priority based scheduling semantic of the run queue is preserved.

15. (Previously presented) A system as recited in claim 8, wherein the processor is further configured to execute computer program instructions for:

inserting a root thread of the second plurality of threads into the run queue;

removing the root thread from the run queue for execution; and

responsive to removing the root thread and independent of any additional other queue access, inserting a next thread of the second plurality of threads into the run queue.

16. (Currently amended) A computer-readable storage medium comprising ~~computer-executable~~ computer-program instructions to manage a run queue of executable threads sorted with respect to one another based on thread priority, the ~~computer-executable~~ computer-program instructions ~~comprising instructions~~ being executable by a processor for:

in a deterministic amount of time that is independent of the number of threads in a second plurality of threads that is priority sorted, the deterministic amount of time being a time to insert a single thread into the run queue, associating the second plurality of threads with a first plurality of threads in the run queue in a manner that maintains a priority based scheduling semantic of the run queue.

17. (Currently amended) A computer-readable storage medium as recited in claim 16, wherein the second plurality of threads comprises a root thread that is operatively coupled to one or more other threads of the second plurality of threads, and wherein the computer-program instructions for associating further comprise instructions for: inserting only the root thread into the first plurality of threads.

18. (Previously presented) A computer-readable storage medium as recited in claim 16, wherein the first plurality of threads is a first linked list data structure, the second plurality of threads is a second linked list data structure comprising a root node that is operatively coupled to one or more other threads in the second plurality of threads, and the deterministic amount of time is a result of a single insert operation to insert the root node into the first linked list data structure.

19. (Currently amended) A computer-readable storage medium as recited in claim 16, wherein the computer-program instructions for associating further comprise instructions for:

inserting only a root thread of the second plurality of threads into the first plurality of threads;

and wherein the ~~computer-executable~~ computer-program instructions further comprise instructions for:

removing the root thread from the run queue; and

responsive to removing the root thread, inserting a next thread of the second plurality of threads into the first plurality of threads in a manner that maintains a priority based scheduling semantic of the run queue .

20. (Currently amended) A computer-readable storage medium as recited in claim 19, wherein the computer-program instructions for inserting the next thread are performed independent of an other queue.

21. (Currently amended) A computer-readable storage medium as recited in claim 16, wherein the computer-program instructions for associating further comprise instructions for:

inserting a root thread of the second plurality of threads into the first plurality;

removing the root thread from the first plurality of threads for execution; and

responsive to removing the root thread, inserting a next thread of the second plurality of threads into the first plurality of threads independent of any additional access to another different queue.

22. (Canceled).

23. (Currently amended) A computer-readable medium comprising computer-program instructions executable by a processor for:

managing a run queue with a run queue data structure, the run queue data structure comprising:

a first dimension data field comprising a first plurality of threads sorted with respect to thread priority; and

a second dimension data field comprising a second plurality of threads sorted based on thread priority, the second plurality of threads comprising a root thread and one or more other threads.

24. (Canceled).